１１．新しい変数を作る

社会調査データを分析する場合に、素データの変数をそのまま使うだけでなく、ある変数のカテゴリーをまとめなおしたり、複数の変数を組み合わせて新しい変数や尺度を作ったりすることがしばしばあります。このような場合、compute, recode, if などのコマンドを使います。

11.1 再コード（recode）による新しい変数の生成

ひとつの変数の値をいくつかまとめて、カテゴリーを作り直す場合に、recode コマンドが便利です。ただし、recode コマンドをそのまま使うと、変数名を変えずに、値の区切り方だけが変わってしまうので、もとのデータが SPSS データファイル上からは、失われてしまいます。

達人方式では、素データファイルが残っているので、それでもかまわないのですが、やっぱり混乱のもとなので、新しい変数名を compute コマンドを使って用意しておく方がよいでしょう。

標準的な書式は、

```
compute 新変数名＝旧変数名.
recode 新変数名(旧値, 旧値 ＝新値)(旧値、旧値＝新値).
```

となります。

たとえば、回答者の年齢を５歳刻みで示す q42a という変数をもとに、年齢 10 歳刻みの新しい変数 age10 を作る場合、つぎのようになります。

```
compute age10=q42b.
recode age10 (1,2=1) (3,4=2) (5,6=3) (7,8=4)
(9,10=5).
value labels age10 1 "20-29" 2 "30-39" 3 "40-49" 4 "50-59"
5 "60-69".
```

１行目の compute は、新しい変数 age10 を用意して、まず年齢５歳刻みの q42b と同じ値を与えます。この段階では age10 は q42b と同じ内容（値）をもっています。

２行目の recode で、1（20-24 歳）と 2（25-29 歳）は 1 に、
3（30-34 歳）と 4（35-39 歳）は２に、
....という具合に、値が組み替えられます。q42b に欠損値が指定されていれば、欠損値はそのままシステム欠損値として引き継がれます。システム欠損値は、欠損値解除ができませんので注意してください。

４～５行目は、新しくこしらえた変数 age10 に値ラベルを貼るコマンドです。必要ならば、変数ラベルもつけておくとよいでしょう。

11.2　算術式による新しい変数の生成（compute）

上の例で用いた、compute コマンドは、**新変数＝(既存の変数を用いた)算術式**によって、新しい変数を生成します。

算術式に用いることのできる記号は、以下の通りです。

	例
加法（足し算）：＋	compute nx1=v1+v2.
減法（引き算）：－	compute nx1=5-v1.
乗法（かけ算）：＊	compute nx1=v1*v2.
除法（わり算）：／	compute nx1=v1/v2.

また、() を使って演算の優先関係を示すことが可能で、カッコは何重にも使えます。
この他にエクセルのような関数を使うこともできます。主な関数は以下の通り。

	例
平方根：SQRT	compute nx1=SQRT(v1).
自然対数：LN	compute nx1=LN(v1+1).
常用対数：LG10	compute nx1=LG10(v1+V2+1).
指数関数：EXP	compute nx1=EXP(v1).
絶対値：ABS	compute nx1=ABS(v1-v2).

わかりやすくするために、関数部分を大文字にしてありますが、実際には大文字と小文字の区別はありません。

compute を使った変数変換の実例

以下では、compute コマンドを使った変数変換の実例をいくつか紹介します。

例１　ID 番号の生成。１桁の地点番号 chiten と３桁のサンプル番号 sample を組み合わせてサンプル全体の ID 番号を生成する（→ 9.2 参照）。

```
compute ID=chiten*1000+sample.
```

例２　合計値の生成。同居きょうだい数 q13a からその他のきょうだい数 q13f までの６つの変数を合算して、きょうだい数という新しい変数を生成する（欠損値は 99。最後に変数ラベルをつけている）。

```
missing values q13a to q13f (99).
compute siblings=q13a+q13b+q13c+q13d+q13e+q13f.
variable labels siblings "きょうだい数".
```

例３　コードの逆転。q24i は「夫婦は同じ姓を名乗った方が良い」に関する意識を尋ね

る項目で、１．賛成、２．やや賛成　３．やや反対　４．反対となっている。これを、点数が高いほど肯定的になるように４、３、２、１となる変数 v24i を生成する（ただし欠損値９はシステム欠損値とする)。(do repeat の項も参照)。

```
missing values q24i(9).
compute v24i=5-q24i.
```

例４　対数変換。30 分から１時間の友人数 q18b と１時間から２時間の友人数 q18c をあわせて「中距離友人数」を計算し、これに１を加えて常用対数に変換する（欠損値は 99。中距離友人数が０になる場合に対数変換後の値が０になるように１を加えてある。最後に変数ラベルをつける)。

```
missing values q18b q18c (99).
compute lfrd302h=lg10(q18b+q18c+1).
variable labels lfrd302h "中距離友人数".
```

11.3 論理式による変数変換if

特定の条件を満たす場合にだけ、特定の変数変換を施したい場合には if コマンドを使います。一般的な書式は、以下のように if の後のカッコ内に論理式をおき、カッコの外に算術式をおきます。

if（論理式）算術式.

論理式のシンタックスは、以下の通りです。算術式とは異なるので注意が必要です。また、論理式の優先関係を示すカッコ（）は何重にも使えます。

関係演算子
＝なら eq
＜なら lt
＞なら gt
≦なら le
≧なら ge
≠なら ne

論理演算子
「かつ」なら and
「または」なら or

if 文を使った変数変換の実例

例 1　特定の値を欠損値に繰り入れる例。婚姻状態 q1 が「1.未婚」である場合、配偶者の両親の所在地 q10 は「8.配偶者はいない」に割り当てます。これは本来データクリーニングで素データファイルを修正すべきところを、プログラム上で一括して修正しています。

```
if (q1 eq 1) q10=8.
```

例 2　職業コードをまとめて、新しい職業変数を作る例。ここで q48 は職種です。ここでは、つぎのようにまとめ直して新しい変数 ocp をつくりたい。

職種 q48	→	職業 ocp
1.専門職		1. 専門・管理・技術職
2.管理職		〃
3.事務職		2. 事務職
4.販売・営業職		3. 販売・サービス職
5.サービス職		〃
6.技能職・労務職		4. 技能・労務・農林職
7.保安職		〃
8.農林漁業従事者		〃
9.その他		9.欠損値コード
0.（欠損値コード）		〃

プログラム上は、まず compute コマンドで ocp ＝ 0 を用意します。
ついで if コマンドで条件にかなうものについてひとつずつ値を指定していきます。

```
compute ocp=0.
if (q48 eq 1) ocp=1.
if (q48 eq 2) ocp=1.
if (q48 eq 3) ocp=2.
if (q48 eq 4) ocp=3.
if (q48 eq 5) ocp=3.
if (q48 eq 6) ocp=4.
if (q48 eq 7) ocp=4.
if (q48 eq 8) ocp=4.
if (q48 eq 9) ocp=9.
if (q48 eq 0) ocp=9.
value labels ocp 1 "専門・管理・技術" 2 "事務" 3 "販売・サービス"
4 "技能・労務・農林" 9 "欠損値コード".
```

例３　雇用形態 q44 と従業先の規模 q45 を合成して、従業上の地位という変数を生成する例、ここでは、雇用形態の「自営業」と「経営者」を従業規模を基準に再分類して、従業員４人以下なら「自営業」、５人以上なら「経営者」として、自営業と自由業を同じカテゴリーにまとめ直すことを意図しています。

まず１行目で新変数 emp をとりあえず q44 と等置します。次に２行目で「自営業」(q44 eq 1)で「従業員１人のみ」(q45 eq 1) と「従業員４人以下」(q45 le 3) の場合を「自営業・自由業」とします。３行目では、「経営者」(q44 eq 4)で「５人以上の業主」((q45 ge 4) and (q45 le 8))を「経営者」としています (q45 eq 9 は欠損値であるため上限を押さえていることに注意)。同様に、５行目以降は、q45 の「経営者」について同じ基準で分類し直しています。

```
compute emp=q44.
if ((q44 eq 1) and ((q45 ge 1) and (q45 le 3))) emp=1.
if ((q44 eq 4) and ((q45 ge 1) and (q45 le 3))) emp=1.
if ((q44 eq 1) and (q45 ge 4)) emp=2.
if ((q44 eq 4) and (q45 ge 4)) emp=2.
if (q44 eq 2) emp=3.
if (q44 eq 3) emp=4.
if (q44 eq 5) emp=5.
if (q44 eq 6) emp=6.
if (q44 eq 7) emp=7.
if (q44 eq 8) emp=8.
variable labels emp "従業上の地位".
value labels emp 1 "自営業主" 2 "経営者" 3 "家族従業者" 4 "正社員" 5 "嘱託・派遣"
6 "ﾊﾟｰﾄ･ｱﾙﾊﾞｲﾄ" 7 "自由業" 8 "無職・学生・主婦".
```

11.4　変数変換の繰り返し do repeat/end repeat

同じタイプの変数変換を多くの変数について行う場合、do repeat/end repeat コマンドを使ってまとめて実行することができます。いくつかの例を紹介します。

例１　コードの逆転。q24a ～ q24i までの 9 つの変数について、１，２，３，4 の値をそれぞれ、４，３，２，１に逆転させる。

```
missing values q24a to q24i (9).
do repeat q24=q24a to q24i
/v24=v24a v24b v24c v24d v24e v24f v24g v24h v24i.
compute v24=5-q24.
end repeat.
```

１行目は、たんに欠損値を指定しているだけです。

２行目はふたつのマクロ変数を定義しています。q24 はここで新たに定義されるマクロ変数で、既存の q24a から q24i を指します。v24 も新たに定義されるマクロ変数で、v24a v24b ....v24i という新変数を指します。新変数は定義されていないので to 表記を使うことができません。新しい変数の名称は何でもよいのですが、q24 が 9 つの変数を指示してますので、新変数を 9 つ用意しなければなりません。

３行目は、マクロ変数を使った compute コマンドです。9 つの変数の組について順次、この計算式による変数変換が施されます。

４行目は、マクロ変数によるルーチンを終了させることを意味しています。

例２　つぎは、if 文による変数変換を繰り返す例。q44 が 8 の場合、q50_1 から q50_14 を一律に 8（非該当）とする一括処理です。

```
do repeat q50=q50_1 to q50_14.
if (q44 eq 8) q50=8.
end repeat.
```

11.5　変数変換を行う場合の作業上の注意点

recode, compute, if, do repeat などを使って、変数変換を行う場合、必ず事前に、変数変換の対象となる既存変数について度数分布を出し、欠損値の様態などを把握しておきましょう。

また変換後は、新変数の度数分布や、旧変数とのクロス集計などを出して、変換がうまくいっているかどうか確認しましょう。